Naval Research Laboratory
Stennis Space Center, MS 39529-5004

NRL/MR-MM/7430--03-8294

A Procedure to Edit Deep-Towed Navigation Data

Charles Mégnin
Warren T. Wood
Dennis A. Lindwall
Joseph F. Gettrust

Seafloor Sciences Branch
Marine Geosciences Division

February 28, 2003

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
February 28, 2003	Memorandum Report	October 1, 2002-January 8, 2003

4. TITLE AND SUBTITLE

A Procedure to Edit Deep-Towed Navigation Data

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
0601153N

6. AUTHOR(S)

Charles Mégnin, Warren T. Wood, Dennis A. Lindwall, and Joseph F. Gettrust

5d. PROJECT NUMBER
74-7814-03

5e. TASK NUMBER

5f. WORK UNIT NUMBER
74-7814-00

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Marine Geosciences Division
Stennis Space Center, MS 39529-5004

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR-MM/7430--03-8294

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

10. SPONSOR / MONITOR'S ACRONYM(S)
ONR

11. SPONSOR / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

While data acquired from deep-towed seismic surveys offer the potential of greatly increased image resolution over the surface-towed approach, accurate positioning constitutes an important challenge which, if not performed accurately, could offset the benefits gained by the proximity between the source and the ocean bottom. We present here a new procedure that optimally determines relay and receiver positions at all times, using the Long Baseline acoustic navigation system. Navigation and seismic data were acquired during the October 2002 Cascadia Margin cruise.

15. SUBJECT TERMS

DTAGS; Deep Towed Acoustics/Geophysics System, Velocity images; Seismic reflection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Joseph Gettrust
Unclassified	Unclassified	Unclassified	UL	37	19b. TELEPHONE NUMBER *(include area code)* (228) 688-5475

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

A Procedure to Edit Deep-Towed Navigation Data

Abstract

While data acquired from deep-towed seismic surveys offer the potential of greatly increased image resolution over the surface-towed approach, accurate positioning constitutes an important challenge which, if not performed accurately, could offset the benefits gained by the proximity between the source and the ocean bottom. We present here a new procedure that optimally determines relay and receiver positions at all times, using the Long Baseline acoustic navigation system. Navigation and seismic data were acquired during the October 2002 Cascadia Margin cruise.

1 Introduction

The Methane Hydrates group at the Naval Research Laboratory has used the broadband Deep Tow Acoustics/Geophysics System (DTAGS) to achieve short length scale resolution of subsurface features [1]. This system offers several advantages over conventional marine acquisition methods due to the proximity of the instrument to the seafloor. In particular, it reduces the size of the Fresnel zone, resulting in better lateral resolution, and enhances the effective source level by reducing the effect of spreading loss.

Accurate location of the source and receiver positions is a necessary condition to the derivation of high-quality velocity images from seismic reflection data. While accurate positioning of the ship and course is readily provided by the on-board Global Positioning System (GPS), such measurements are not available for deep-towed instruments so that the geographical position of the moving instrument at depth must be derived acoustically via an attached acoustic relay. This is determined through a two-step process whereby the positions of four bottom transponders are established and later used to triangulate the position of the acoustic relay.

The DTAGS instrument is typically operated at a depth close to the ocean bottom, located at some 1,000 to 2,000 meters of cable length from the ship. It is connected to an array of 48 receivers at distances between 20 and 450

meters from the source. In deep water and using large towing distances, such navigation data is characterized by the presence of a large proportion of erroneous positions (possibly greater than 75% of the total number of readings), which is likely caused by cycle-skipping of the transponder pulse, and the presence of air bubbles below the hull. Because the instruments whose position we seek move very slowly (typically two knots), accurate positions can be found if the navigation fixes can be edited effectively. Here, we describe a new and simple method used to edit the navigation data acquired during the October 2002 Casacadia Margin cruise. This methodology helps detect and discard incorrect geographical positions. Accurate positioning of the entire track-line is then obtained by interpolating the missing fix positions between the remaining locations. This approach is justified over areas where the ship follows a steady course at relatively constant heading and speed. Its position at any given time can therefore be predicted with excellent accuracy from the preceding and following positions.

In the following we define a *'fix'* as a navigation datum recording event as it applies either to the ship or to the acoustic relay's position. We record such fixes every 5 seconds or so. There are far more fixes than *'energy points'* (*ep*'s) which we define as seismic data events, typically recorded every 30 seconds.

2 Method

Acoustic positioning of the acoustic relay is obtained from a two-step process: first, four or more anchored transponders are placed on the seafloor surrounding the region of interest. Their position at the bottom is accurately determined by continuously measuring the travel time of sound waves of four different frequencies to an hull-mounted transponder as the ship follows a clover-like path around and between the array elements. A least square inversion of the travel times, making use of the measured ocean sound velocity profile and the ship GPS data, yields their positions on the ocean bottom. Second, the position of the moving acoustic relay at depth is derived by continuous triangulation, measuring the travel times between relay and bottom transponders. Both steps are performed using the WinFrog^{TM} software and a detailed explanation of this procedure is given in [2]. We expect the location

of the relay fixes to be most accurate within the rectangle circumscribed by the bottom transponders and to diminish with increasing distance from that rectangular area.

Accurate positioning of the acoustic relay can only be achieved if the ship position is known with great accuracy. Fortunately, over 90% of GPS-derived ship fixes appear reasonable (Figure 1) and the editing of these data thus lends itself well to automation. Figure 1 (derived using the Generic Mapping Tool (GMT) script [3] *plot_course.gmt* – Appendix C.3.1) is a typical example of raw ship navigation data. The GPS positions are represented by the red trace and appear to be in good agreement with the actual trace of the ship. The course of the ship is the shortest path among fixes (blue numbers), and the incoherent fixes resulting from the few glitches are prominent and readily removed by visual inspection. To do so, it is easiest to look for abrupt changes in northing and easting derivatives such as Figure 2 (*plot_jumps.gmt* – Appendix C.3.3) which displays latitudinal (blue trace) and longitudinal (red trace) derivatives of the ship's position ($\frac{\Delta \text{lat}}{\Delta \text{fix}}$ and $\frac{\Delta \text{lon}}{\Delta \text{fix}}$). The *flag_jumps.tcsh* shellscript (Appendix C.1.4) can be used simultaneously to flag all changes in easting and northing greater than a specified distance (all code and scripts are included Appendix C). Figure 3 is a typical example of the quality of raw acoustic relay navigation data, where the large majority of fix locations are spurious. The blue trace is that of the ship after removal of the questionable fixes and interpolation. The red trace joins the successive acoustic locations of the relay. Figure 4 (*plot_distaz.gmt* – Appendix C.3.2) shows the unedited relay position relative to the edited position of the ship. The top Figure shows distance and azimuth in (r, θ) space. The middle plot shows the distance between instruments as a function of the fix number. The red line is an estimate of this value, derived from manually recorded logs of relay depth from pressure sensors located on the towed system and length of cable out. It is used as a discriminant in the case where two or more plausible relay locations are inferred (*e.g.*between fixes # 200 and 375). Finally, the bottom plot represents the azimuth of the ship to the acoustic relay. In the absence of strong lateral currents, the value along the ship's straight course should remain close to 0° or 180° $\pm$ 20° and we also expect abrupt changes to be caused by erroneous data (*e.g.*near fix # 260). A complementary tool which displays the northing and easting derivatives of the acoustic relay is shown in Figure 5.

The numerous incoherent positions must be manually removed by visual inspection. In the absence of a prominent baseline of correct relay positions (as is the case for the GPS data), flagging with the help of *flag_jumps.tcsh* is likely to yield unreasonable results. Figure 6 shows the predicted course of the ship (red line) and the relay (green line) after application of the editing procedure. The predicted relay fix positions are shown with the blue stars. Note that most are evenly spaced, as expected from the steady ship's velocity. Departure from the constant spacing is caused by changes in relay depth. Figure 7 shows the equivalent representation in (θ, r) space (blue line). The red line on the distance plot is derived from the on board depth logs.

Errors in the final estimate of the position of the acoustic relay are directly related to inaccuracies in the building blocks of this method: the ship's position and the positions of the bottom transponders. Although a formal error analysis is beyond the scope of this report, the GPS-derived ship's position has a sub-meter accuracy and the error estimate from the WinFrog^{TM} inversion for the transponder locations is on the order of 1 to 2 meters. Assuming that these errors accumulate constructively, a worst-case estimate of the absolute error of the relay position at each fix is less than 10 meters.

3 Conclusion

Because the towing ship follows a predictable path, the straightforward procedure described above correctly interpolates a large number of fix locations from a possibly small number of positions deemed accurate. Although this semi-automated procedure is effective along a great-circle line, it is not appropriate for locations following a change in heading. This is typically not a problem as ship turns occur far from the region of geological interest.

Acknowledgements: This project was funded by NRL-SSC code 7400, program number 0601153N.

For additional information please contact:
Charles Mégnin
NRL Code 7432

cmegnin@nrlssc.navy.mil
Voice: 228-688-2483
Fax: 228-688-5752

References

[1] W. T. Wood and J. F. Gettrust. New developments in deep-towed seismic acquisition. *Oceans 2002 MTS/IEEE Conference Proceedings*, pages 1,139–1,142, 2002.

[2] D. A. Lindwall, W. T. Wood, C. Mégnin, and J. F. Gettrust. Acoustic transponder navigation using winfrog as used on the Cascadia Margin cruise; Bull's Eye site, October 2002. Memorandum Report, Naval Research Laboratory, in preparation.

[3] P. Wessel and W. H. F. Smith. New, improved version of the Generic Mapping Tool released. *EOS Trans. AGU*, **72**, 441:445–446, 1991.

A Summary of procedure

This chapter summarizes the steps necessary to edit the ship and relay navigation data (A.1) and to merge the results with the seismix data (A.2). A worked out example with actual data is given in section B.

A.1 Determination of the fix location

- Split the WinFrog navigation file *.DAT with *winfrog_parse.tcsh* (Appendix C.1.2) to produce a file for each instrument:
  - *.ship.raw – the ship navigation data
  - *.fish.raw – the relay navigation data

  Then, for each of the two resulting navigation files,
- Remove the repeat fixes with *remove_repeats.tcsh* (Appendix C.1.3).
- Flag and remove fixes that are further than a given distance from the previous fix. The cutoff distance is readily obtained from the ship speed an the rate at which data fixes are recorded. For ship data, use *flag_jumps.tcsh* (Appendix C.1.4) to detect the anomalous fixes and remove manually.
- Flag and remove fixes whose northing and easting disagree with the heading of the ship. (Use *flag_course.tcsh* (Appendix C.1.5) and remove manually).
- Interpolate among the remaining fixes with *interpolate.tcsh* (Appendix C.1.6).
- Visual inspection of the navigation data is provided with three GMT scripts:
  - *plot_course.gmt* displays the courses of the ship and relay (Appendix C.3.1). See Figures 1 and 6.
  - *plot_jumps.gmt* plots latitudinal and longitudinal derivatives of the navigation data (Appendix C.3.3). See Figures 2 and 5.
  - *plot_distaz.gmt* displays azimuth and distance from the ship to the relay (Appendix C.3.2). See Figures 4 and 7.

- Ship and relay data files can be merged with *merge_ship_fish*.tcsh (Appendix C.1.7).

A.2 Updating the SEGY headers

Source and receiver locations are presently placed in the segy headers through the process summarized below:

1. Incorporate the time information from the .DAT file into the navigation file with *2nav.tcsh* (Appendix C.1.9) produces a .nav file (this step could have been incorporated in *merge_ship_fish.tcsh*).
2. Convert segy file to su format using *segy2su.tcsh* (Appendix C.1.8).
3. Compute the position of the ship and the relay at the time of the shot by doing an interpolation of the fix time with *process_date.tcsh* (Appendix C.1.10).
4. Specify the position of each receiver along the line relative to the source with *create_channel_data.tcsh* (Appendix C.1.12).
5. update the positions in the su headers with *update_positions.tcsh* (Appendix C.1.13).
6. convert su file to segy *su2segy.tcsh* (Appendix C.1.14).

B Example

In this section, we present a detailed example of the editing procedure summarized in section A used on the line *CM5* during the Cascadia cruise. All data is provided on the CD under the directory *examples*.

B.1 Preliminary editing

The raw Winfrog navigation data is saved in the file *cm05.DAT* and the segy file is *CM5_1996*.
First, extract the relevant navigation data from the Winfrog file (C.1.2):

```
winfrog_parse.tcsh cm05.dat
```

produces the two raw navigation files *cm05.fish.raw* and *cm05.ship.raw*.
Remove the repeat fixes for the ship and the relay traces (C.1.3):

```
remove_repeat.tcsh cm05.ship.raw cm05.ship.junk
remove_repeat.tcsh cm05.fish.raw cm05.fish.junk
```

and interpolate the missing fixes:

```
interpolate.tcsh cm05.ship.junk cm05.ship.0
interpolate.tcsh cm05.fish.junk cm05.fish.0
```

B.2 Editing the ship GPS data

Identify the fixes that are more than 50 meters off of the previous fix (C.1.4):

```
flag_jumps.tcsh cm05.ship.0 cm05.ship.jumps 50
```

The flagged fixes are in the file *cm05.ship.jumps* and are removed manually from *cm05.ship.0* and saved in the file *cm05.ship.nojumps*. Note that not all fixes flagged should be removed as one excursion from the course will flag two fixes: the one off course and the following one, for example if it is back on course. The identified fixes must therefore be inspected individually before removal. It is also advisable to repeat this procedure on *cm05.ship.nojumps*. Identify the fixes whose northings and eastings disagree with the ship's course. For the present example, northing and eastings are negative roughly up to fix number 680. Therefore (C.1.5):

```
flag_course.tcsh cm05.ship.nojumps cm05.ship.offcourse 0 0
```

The flagged fixes are in the file *cm05.ship.offcourse* and are removed manually from *cm05.ship.nojumps* and saved in the file *cm05.ship.oncourse*. Here also, flagged fixes must be inspected individually before removal and flag_course.tcsh should be repeated as many times as necessary to remove all off course fixes. Again, remove possible repeat fixes with (C.1.3):

```
remove_repeat.tcsh cm05.ship.oncourse cm05.ship.oncourse.norepeats
```

Finally, interpolate the missing fixes with (C.1.6):

```
interpolate.tcsh cm05.ship.oncourse.norepeats cm05.ship.final
```

All through this process, the results should be visually inspected using the *plot_course.gmt* GMT script. To show the clean trace for line# 5 and display every 100th fix (C.3.1):

```
plot_course.gmt cm05.ship.clean 5 ship 100
```

For this example, *plot_course.gmt* was executed at each intermediate step, generating the postcript files *course.ship.5.*.ps*.

B.3 Editing the relay data

The relay data is processed the same way as the ship data but because it is considerably noisier, the processing is somewhat more time consuming. The file *cm05.fish.final* is the result of the processing described in section B.2.
The two GMT scripts *plot_distaz.gmt* and *plot_jumps.tcsh* simplify the task of identifying the erroneous fixes.
To display northing and easting jumps (Appendix C.3.3):

```
plot_jumps.tcsh cm05.fish.final
```

generates the postcript file cm05.fish.jumps.ps
To create a distance/azimuth plot, first merge ship and relay navigation files (Appendix C.1.7):

```
merge_fish_ship.tcsh cm05.ship.final cm05.fish.final cm05.final
```

the output file *cm05.final* will be fed to the GMT script (Appendix C.3.2):

```
plot_distaz.gmt cm05.final 5
```

to generate the postscript distaz.5.ps. On the distance plot, the black line corresponds to the edited data and the red line to the on-board log data in the file dist.est.
Note that ep bounds can be specified for both *plot_distaz.gmt* and *plot_jumps.tcsh*. To generate a distance/azimuth plot between energy points 754 and 986, (Appendix C.3.2):

```
plot_distaz.gmt cm05.final 5 754 986
```

and likewise with *plot_jumps.tcsh*.
Incorporate the time data from the Winfrog .DAT file into the merged navigation data (Appendix C.1.9):

```
2nav.tcsh cm05.final cm05.DAT
```

generates the file *cm05.nav* which is the navigation file used to update the information in the segy headers.

B.4 Updating the segy headers

Convert the segy data *CM5_1996* to su format on a little-endian machine such as a Linux box (Appendix C.1.8):

```
segy2su.tcsh CM5_1996 1
```

generates the su data file *CM5_1996.su*.
Compute the position of ship and relay energy points time by interpolating the ep time from the fix times. Any active channel (here 48) can be passed as a parameter (Appendix C.1.10):

```
process_date.tcsh CM5_1996 48 cm05.nav
```

creates a new navigation file cm05.nav.out
Specify the position of each receiver along the line relative to the DTAGS relay (Appendix C.1.12):

```
create_channel_data.tcsh
```

generates the file *channel.dat*.

Generate a *baseline.dat* file with two entries: base longitude and base-latitude (in this order). All northings and eastings in the headers will be computed relative to the geographic point specified.
Compute the receiver positions and write source and receiver positions into the su file headers (Appendix C.1.13):

```
update_positions.tcsh CM5_1996 cm05.nav.out channel.dat baseline.dat
```

generates an su file with the navigation headers *CM5_1996.level=1.su*.
Finally, convert *CM5_1996.level=1.su* back to segy format:

```
su2segy.tcsh CM5_1996.level=1 1
```

C Source code usage

All code and dependencies are included in the attached CD (contents in section D).
All shellscripts (section C.1) and GMT code (section C.3) are written using the Unix/Linux scripting language tcsh.

C.1 shellscripts

C.1.1 env.tcsh

Description: Contains the definition of the system-dependent variables. This file should be edited to reflect the location of the various parameters on the users desktop.

Usage:

```
source env.tcsh
```

Dependencies: none

C.1.2 winfrog_parse.tcsh

Description: This shell creates two navigation files from the winfrog *.DAT file, one for the ship and one for the relay :

- *.ship.raw
- *.fish.raw

The output format is:
line #, fix #, latitude, longitude

Usage:

```
winfrog_parse.tcsh [winfrog navigation file]
```

Dependencies: None

C.1.3 remove_repeats.tcsh

Description: This shell removes consecutive identical fixes (but preserves the first).

Usage:

```
remove_repeats.tcsh [myfile.nav][myfile.out]
```

Dependencies: None

C.1.4 flag_jumps.tcsh

Description: Identifies fixes whose distance from the previous fix is greater than the tolerance level.

Usage:

```
flag_jumps.tcsh[navigation file][output file][tolerance (meters)]
```

Dependencies:

- dist_az.c (Appendix C.2.3).

C.1.5 flag_course.tcsh

Description: Identifies fixes that do not follow the northing and easting specified.

Usage:

```
flag_course.tcsh[navigation file][output file]
                [1: increasing lat/0:decreasing lat]
                [1: increasing long/0:decreasing long]
```

Dependencies: None

C.1.6 interpolate.tcsh

Description: This shell interpolates equidistant fixes among navigation fixes. It uses the C-code pathsegments.c (C.2.2) described below.

Usage:

```
interpolate.tcsh[navigation file][output file]
```

Dependencies:

- pathsegments.c (Appendix C.2.2)

C.1.7 merge_fish_ship.tcsh

Description: This shell merges the ship navigation file with the relay navigation into one file. It computes the azimuth of the relay from the ship and the distance between them. These values are calculated with the use of the C-code sphdist.c (C.2.5) and azmth.c (C.2.4) below.

Usage:

```
merge_fish_ship.tcsh [ship file] [fish file][output file]
```

Dependencies:

- sphdist.c (Appendix C.2.5).
- azmth.c (Appendix C.2.4).
- check_file_exists.tcsh (Appendix C.1.15).

C.1.8 segy2su.tcsh

Description: Converts a segy file to su format. The output file name is the input file with ".su" appended so that the command:
segy2su.tcsh CM5
will create the su format file:
CM5.su

Usage:

```
segy2su.tcsh[segy file]
```

Dependencies:

- su

C.1.9 2nav.tcsh

Description: Merges the time information in the Winfrog .DAT file with the output from *merge_fish_ship.tcsh* (Appendix C.1.7).

Usage:

```
2nav.tcsh[navigation file][.DAT file]
```

Dependencies:

- julday.c (Appendix C.2.6)

C.1.10 process_date.tcsh

Description: Computes the position of the ship and the relay at the time when the seismic signal is fired (*i.e.* ep) from the fix data. This is done by interpolating in time the fix positions preceding and following the ep time.

Usage:

```
process_date.tcsh[su file prefix][channel #][navigation file]
```

where the channel can be any (existing) channel and the navigation file is the output from *2nav.tex* (Appendix C.1.9

Dependencies:

- su
- get_su_date.tcsh (Appendix C.1.11)
- fitD.c (Appendix C.2.9)

C.1.11 get_su_date.tcsh

Description: Extract date from su header

Usage:

`get_su_date.tcsh[su file][ep #][channel #]`

Dependencies:

- su

C.1.12 create_channel_data.tcsh

Description: Convenience shell to create the 2-column file with channel number and distance from relay for each active channel.

Usage:

`create_channel_data.tcsh`

creates a channel.dat file. Various parameters may be cahnged as needed:

- offset: source-first receiver offset in meters (21 for Cascadia cruise)
- cutoff: last short spaced channel number (24 for Cascadia)
- lchan: last channel (48 for Cascadia)
- sp1: receiver spacing in meters for narrow-spaced array segment (3 for Cascadia)
- sp2: receiver spacing in meters for large-spaced array segment (15 for Cascadia)
- initi: number of first channel (3 for Cascadia)

Dependencies: None

C.1.13 update_positions.tcsh

Description: Computes and writes the positions of source and relay in su header file. The values written are northings and eastings *IN CENTIMETERS* from a base longitude and latitude.

Usage:

```
update_positions.tcsh[su file prefix][navigation file][channel file][base file]
```

The navigation file is output by *2nav.tcsh* (Appendix C.1.9).
The channel file is output by *create_channel_data.tcsh* (Appendix C.1.12).
The base file contains the position of the point from which northings and eastings are derived: 2 entries - longitude and latitude in this order.

Dependencies:

- su
- GMT
- EastingDriver.c (Appendix C.2.8)
- NorthingDriver.c (Appendix C.2.7)
- check_file_exists.tcsh (Appendix C.1.15)

C.1.14 su2segy.tcsh

Description: Converts an su file to segy format. The output file looses the su extension of the input file so that:
su2segy.tcsh CM5.su 1
creates a little endian CM5 segy formatteed file

Usage:

```
su2segy.tcsh[su file][0:big-endian \ 1:little-endian]
```

Dependencies:

- su

C.1.15 check_file_exists.tcsh

Description: Checks that a file exists.

Usage:

```
check_file_exists.tcsh [filename]
```

Dependencies: None

C.2 C-code

C.2.1 Libraries

Most of the make files for the C-code in this section calls on four libraries:

- libio.a
- libsphere.a
- libstats.a
- libtools.a

the makefiles for these libraries are under the *src* directory under the names libio.mak, libsphere.mak, etc.

C.2.2 pathsegments.c

Description: Computes the latitude and longitude of n equidistant segments between two geographical points.

Usage:

```
pathsegments.bin[navigation file][output file]
```

C.2.3 dist_az.c

Description: Computes distance and azimuth between two points. The makefile can be invoked with the command:

```
make -f Makefile.distaz
```

Usage:

```
dist_az -n [lat pt# 1][long pt# 1] [lat pt# 2][long pt# 2]
```

latitudes and longitudes in degrees.

C.2.4 azmth.c

Description: Calculates the azimuth of a target point (theta_t, phi_t) measured at an obsveration point (theta_o,phi_o); minor-arc is assumed; all in radian; azimuth [-PI,+PI] measured clockwise from north; if the observation point is at at a pole, or if the distance between the two points is 0 or pi, a value of 0 is returned.

Usage:

```
azmth.bin [lat 1][lon 1][lat 2][lon 2][format: 0=theta, phi / 1=lat,long]
```

N.B.: The 5^{th} argument must be 0 if the data is colatitude/longitude and 1 for latitude longitude.

C.2.5 sphdist.c

Description: Calculates the minor-arc distance in radians between two points on the unit sphere.

Usage:

```
sphdist.bin [lat 1][lon 1][lat 2][lon 2][format: 0=theta,phi / 1=lat,long]
```

N.B.: The 5^{th} argument must be 0 if the data is colatitude/longitude and 1 for latitude/longitude.

C.2.6 julday.c

Description: Returns the julian day for a given date. To compile,

```
cc -o julday.bin julday.c
```

Usage:

```
julday.bin [mm][dd][yyyy]
```

C.2.7 northingDriver.c

Description: Computes the northing distance in meters from a base point.

Usage:

`northingDriver.bin[base latitude][base longitude][latitude][longitude]`

dependencies

- flat.c

C.2.8 eastingDriver.c

Description: Computes the easting distance in meters from a base point.

Usage:

`eastingDriver.bin[base latitude][base longitude][latitude][longitude]`

dependencies

- flat.c

C.2.9 fitD.c

Description: A simple routine to linearly interpolate between two points.

Usage:

`fitD.bin [x1][y1][x2][y2][desired x]`

returns the y value at the desired x.

Dependencies None

C.3 GMT-code

C.3.1 plot_course.gmt

Description: Plots a map of the ship and fish courses. A fix number is plotted with the specified periodicity.

Usage:

```
plot_course.gmt[navigation file][line #][fish or ship][periodicity]
```

For example, to plot the raw fish trace from the file cm05.fish.raw (trace # 5) and to display the fix number every 50 fixes: plot_course.gmt[cm05.fish.raw][5][fish][50]

Dependencies:

- GMT
- colors.tcsh (Appendix C.3.4)
- ghostscript

C.3.2 plot_distaz.gmt

Description: Plots Distance and azimuth from the ship to the fish, as well as a distance vs azimuth in (r, θ) space.

Usage:

```
plot_distaz.gmt [navigation file][line #][min fix][max fix][increment]
```

For example, to plot the data from the (merged) file cm05.0 (trace # 5) and to display the fixes 250 to 860, every 60 fixes, try:

```
plot_distaz.gmt cm05.0 5 250 860 60
```

Dependencies:

- GMT
- merge_fish_ship.tcsh *must be run prior to this script* (Appendix C.1.7)
- check_file_exists.tcsh (Appendix C.1.15)
- colors.tcsh (Appendix C.3.4)
- ghostscript

C.3.3 plot_jumps.gmt

Description: This shell computes and plots the azimuthal derivatives of the navigation data (derivatives in latitude and longitude). Values for missing fixes are interpolated before the derivatives are computed. Total derivative (Euclidian norm) can be computed and included in the plot by setting:

```
set amp = 1 # plot amplitude of derivative
```

Usage:

```
plot_jumps.gmt [navigation file] ([min fix] [max fix])
```

*e.g.*to plot azimuthal derivatives from file cm05.ship.raw for all fixes

```
plot_jumps.gmt cm05.ship.raw
```

*e.g.*to plot azimuthal derivatives from file cm05.ship.raw between fixes 112 and 234

```
plot_jumps.gmt cm05.ship.raw 112 234
```

Dependencies:

- GMT
- interpolate.tcsh (Appendix C.1.6)
- colors.tcsh (Appendix C.3.4)
- ghostscript

C.3.4 colors.tcsh

Description: Defines the following set of colors for GMT scripts:
white, black, blue, lightblue, brightred, red, green, yellow, orange, mauve, pink.

Usage:

```
source colors.tcsh
```

Dependencies None

D CD

D.1 Contents

- The code in Appendix C:
  - shellscripts in the *shells* directory
  - gmt shells in the *plots* directory
  - dist_az.c and julday.c in the *other* directory
  - other C-code in the *src* directory
- The Generic Mapping Tool (GMT) v. 3.4.1 in the *gmt* directory (precompiled on RedHat 7.3 and running on 8.0)
- maps to be used by GMT in the *maps* directory
- The netcdf library v. 3 in the *netcdf* directory (precompiled on RedHat 7.3 and running on 8.0)
- Seismic Unix (su) v. 3.6 in the *su* directory (precompiled on RedHat 7.3 and running on 8.0)
- This document in pdf format in the *text* directory

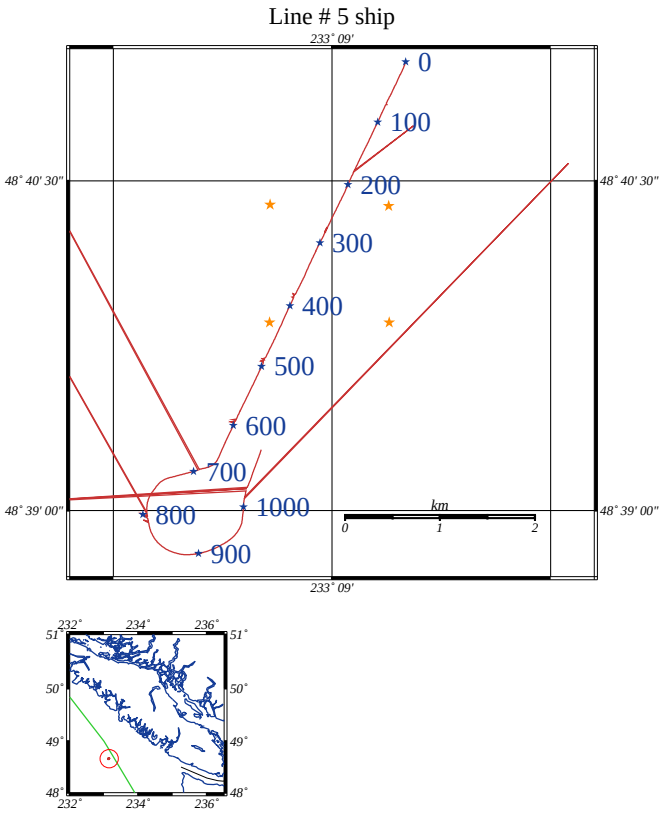

Figure 1: Raw ship navigation data. Top: The location glitches appear as large excursions from the ship's course which is the shortest distance between the fix numbers in blue. The orange stars represent the position of the bottom transponders. Bottom: Location of the survey off of Vancouver Island, B.C.. The boundary between the Juan de Fuca and the North American plates is shown in green. (This figure is made with the GMT script *plot_course.gmt*, Appendix C.3.1).

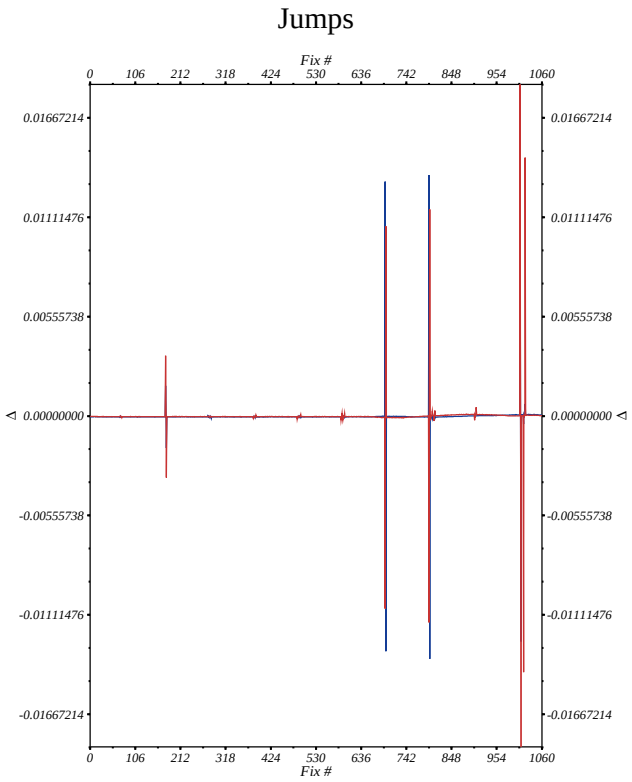

Figure 2: Change in the ship's latitude (blue) and longitude (red), for the course in Figure 1. (This figure is made with the GMT script *plot_jumps.gmt*, Appendix C.3.3).

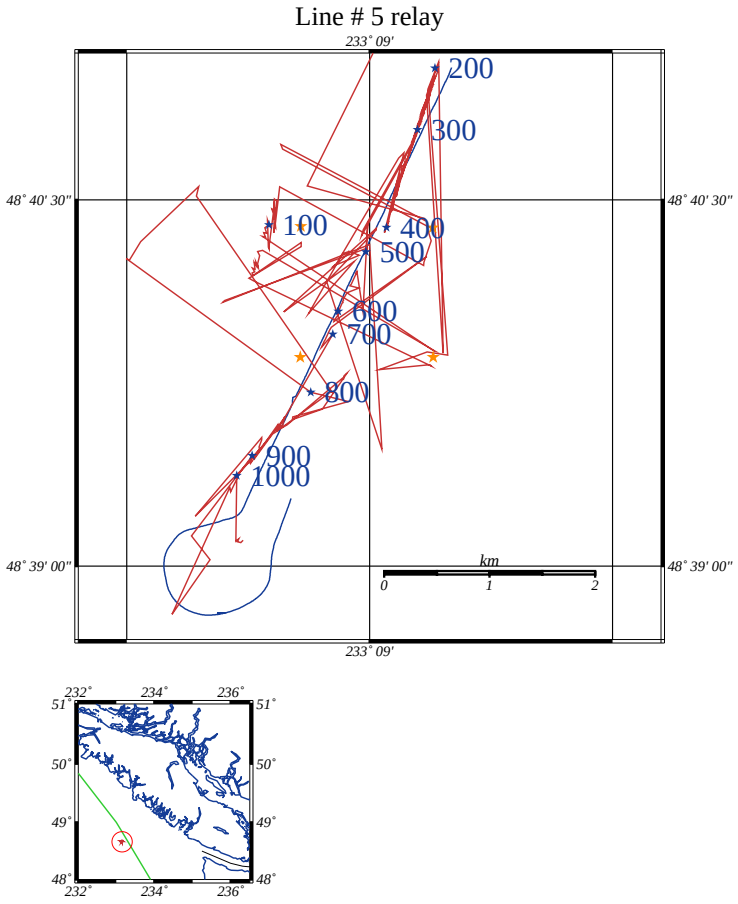

Figure 3: Raw navigation data for the relay. The edited ship's course is shown in blue and the raw relay navigation data in red. The course of the acoustic relay is nearly impossible to follow as most of its fix positions are erroneous. (This figure is made with the GMT script *plot_course.gmt*, Appendix C.3.1).

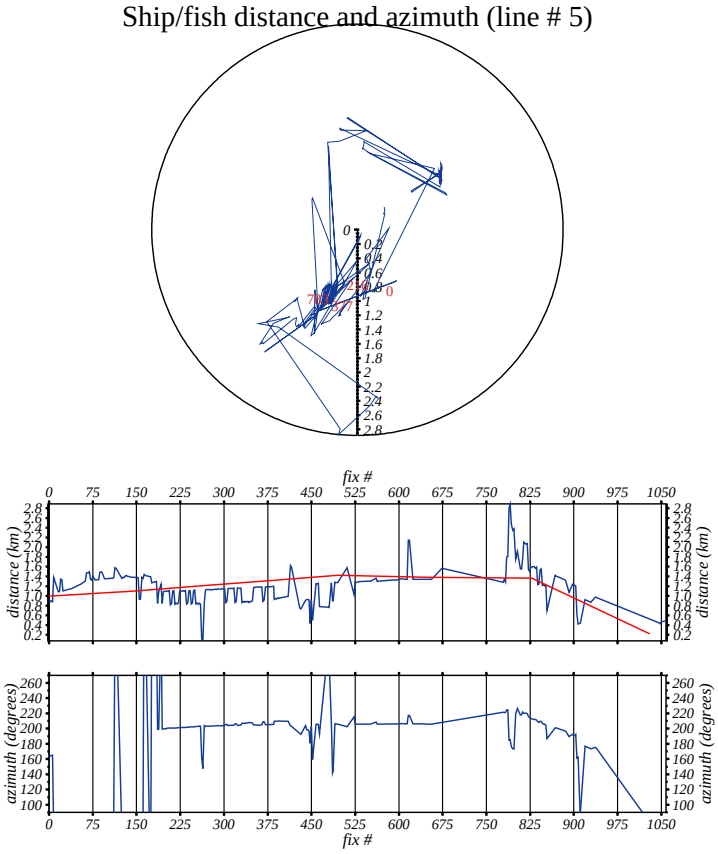

Figure 4: Distance and azimuth plots between ship (edited data) and relay (raw data) corresponding to Figure 3. Top: Fix positions in (r, θ) space. Middle: distance for each fix. The red line corresponds to the distance as calculated from relay depth and length-of-cable logs. Bottom: Azimuth for each fix. The bottom plots allow to rapidly identify off-course positions. (This figure is made with the GMT script *plot_distaz.gmt*, Appendix C.3.2).

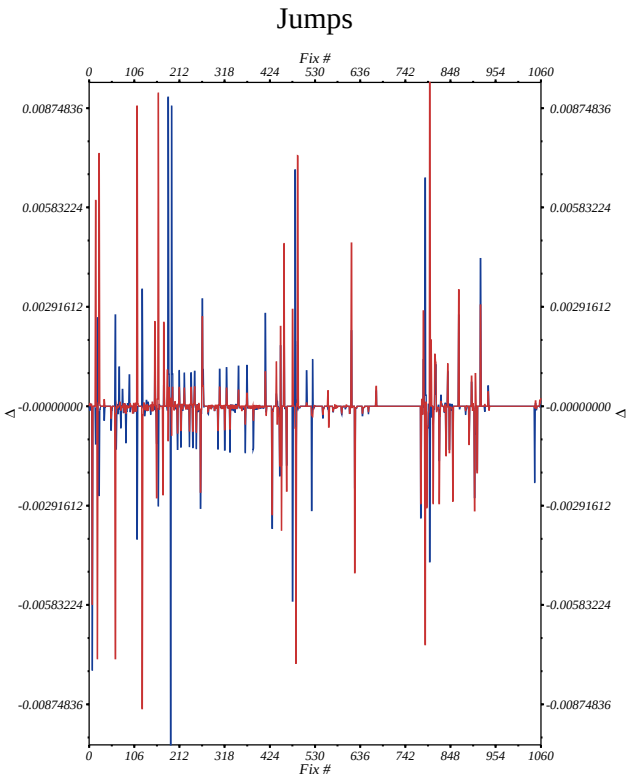

Figure 5: Change in the relay's latitude (blue) and longitude (red), for the course in Figure 3. (This figure is made with the GMT script *plot_jumps.gmt*, Appendix C.3.3).

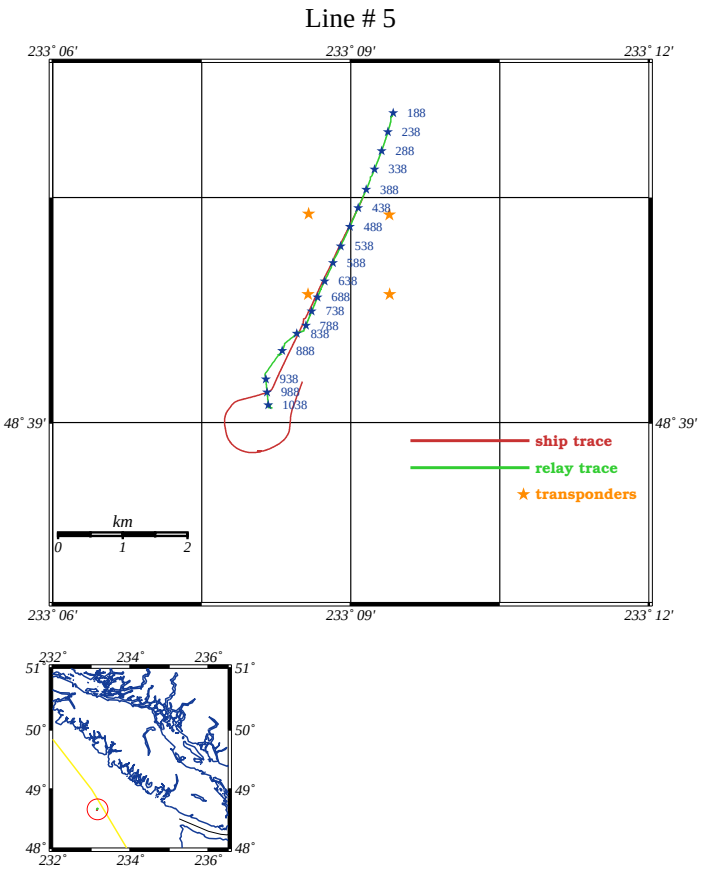

Figure 6: Corrected ship (red line) and relay (green line) positions. (This figure is made with the GMT script *plot_course.gmt*, Appendix C.3.1).

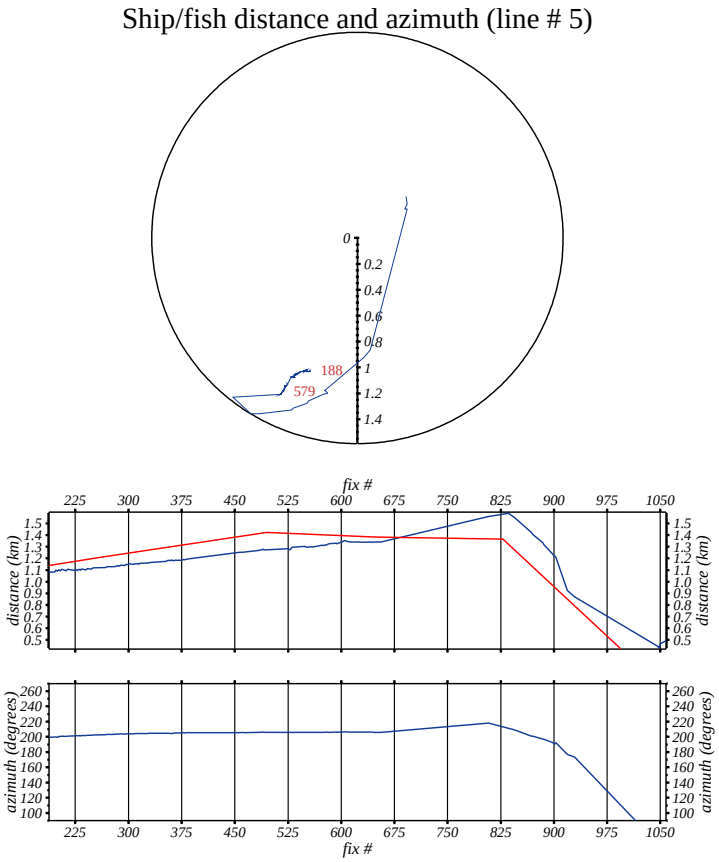

Figure 7: Distance/azimuth plot for the edited data. The red line is derived from the data log and the blue line corresponds to the positions in Figure 6. (This figure is made with the GMT script *plot_distaz.gmt*, Appendix C.3.2).

.